GRADUAAT IN HET
PROGRAMMEREN

Project WPL 1 - Script

Project

1.5

Version

Workplace learning 1 – 2023/24

Course unit

DE HOGESCHOOL MET HET NETWERK

Table of contents

1 Assignment overview

1.1 Purpose of the project

During the WPL1 project weeks, in addition to a C# project, a web application is also developed. The aim is to put the knowledge from the lessons into practice during a long-term assignment.

1.1.1 Web application (CV)

The web application is developed in 2 phases. the First, a CV is drawn up in which: focus is on the desktop version.
In the 2ephase, the CV is expanded so that the website*responsive*is becoming. The website needs to be expanded to also look good on smartphones.

1.1.2 C# application (Cookie Clicker)

The project is based on the game Cookie Clicker. You can find an online version of this gamehere (https://cookieclicker.ee).
The aim of the game is to bake as many cookies as possible as quickly as possible.

1.2 Planning & Phases

The entire application will be implemented in several phases. New functionalities are added to the project in each phase. In this way, the complexity and necessary skills are also built up step by step.

Each phase will also require more and more input from the student.

Timing	Lesson component	Phase	Product
Lesson week 6	Web	Part W.1	Web page CV
Lesson week 7	C# Essentials	Part C.1	Basic version 1 Cookie Clicker
Lesson week 8	Web	Part W.1	Web page CV
Lesson week 9	C# Essentials	Part C.2	Extension version 2 Cookie Clicker

Lesson week 10	Web	Part W.2	Expansion of CV Responsiveness
Lesson week 11	C# Essentials	Part C.3	Extension version 3 Cookie Clicker

So each phase is built in one week WPL1. This means that during the WPL1 session the project is worked on under the guidance of the coaches. However, students are expected to complete this further outside the guided session if they are not ready.

1.3 Submit assignment

Although the project consists of several phases, the result only needs to be submittedafter the last phase . In other words, the entire C# application and the*responsive* web design resume assignment.

See the section on submitting the assignment later in this document.

1.4 Evaluation

Any form of plagiarism will immediately be punished with a 0 on the entire assignment. This includes: copying code indiscriminately from YouTube videos, from fellow students, or from the internet. Or any other form of plagiarism. You write all your code yourself! You can only search for design ideas on the internet. You can convert the design into code yourself!

The distribution key for the Web and C# project are as follows:

Cookie Clicker (C#)	60%
Online CV (Web)	40%

The permanent evaluation for it**C# project**is based on the following criteria:

%	Element	Example	Part
10	Is the requested functionality	*Is the cookie image visible?* *Can the cookie be clicked?* *Is the total amount of cookies displayed correctly?*	Cookie Clicker1

	present and working this correct?	*Are the investments shown?* *Do any unexpected errors occur while playing the game?* *...*	
10	Is the requested functionality present and working this correct?	*Additional:* *Are the purchased investments shown? Does the bonus system work correctly?* *Have the new investments been implemented? ...*	Cookie Clicker2
50	Is the requested functionality present and working this correct?	*Additional:* *Do the animations work correctly?* *Have the gamification principles been applied? ...*	Cookie Clicker3
20	Code quality	*Are the correct programming techniques used? (private variables where necessary, use of methods, ...) Is the code structured and clearly readable?* *Is the code sufficiently documented? ...*	Cookie Clicker3
10	General	*Are all built (extra) functionalities in line with the requirements?* ...	Cookie Clicker3

The permanent evaluation for it**web project**is based on the following criteria:

%	Element	Example	Part
10	Is all content present?	*Is all personal information present? Are any diplomas obtained available? Does the resume have a photo?* *...*	CV1
20	Is the requested functionality present and working this correct?	*Does the table (work experience) have an alternating background color for each line?* *Is it possible to jump from the footer to the beginning of the web page?* *Does the photo get a shadow effect when you move the mouse over it?* *...*	CV1
40	Look & feel + responsiveness	***Does the CV look nice and professional?*** *Does the CV have a clear and readable design?*	CV1

		Do the colors and fonts used match each other? Is your profile photo appropriate for a professional context? ... *Do elements adapt to changes in screen size where necessary?* *Are all elements positioned properly regardless of screen size?* *Do all texts and photos remain readable?* *To what extent can this CV really add value if you were to send it to an internship? Or would they just immediately reject you when they see your results?*	
25	Technical complexity	*Were many different HTML elements, CSS properties, CSS selectors and pseudoclasses used? Have you looked up and implemented something new yourself? Have additional features been added that were not requested? ...* *SHOW HERE WHAT YOU KNOW AND CAN DO*	CV1
5	General	*Are all built (extra) functionalities in line with the requirements, without conflicting with them?* *Is the project presented to the lecturer in the correct format?(**always upload .zip files. –5 on this section if you upload something else**)* *Have code conventions always been taken into account?*...	CV1

NB: The projects are made individually. In case of plagiarism, collaboration or fraud, a failing grade will be awarded.

2 Part W.1 – Web application CV

2.1 Concept(s)

A CV (curriculum vitae) is a document that describes the professional qualities and life history of a person. The document is often used in the first phase of an application. In other words, it is a means to sell yourself as a professional to a potential employer.

2.2 Minimum Requirements

In the first version of your curriculum vitae (CV) website, you will write a desktop version of your CV.
Your CV must contain at least the following information:

- Introduction about yourself. Who are you, what drives you. (See also “Communication Skills” course)
- A photo of yourself (for professional purposes).
- You obtained diplomas.
- Highlight your best skills/knowledge (technical skills and social skills).
- Language skills.
- A list of all your work experience (student jobs, responsibilities within associations, projects completed at home, etc.). Work experiences should not be IT related.

In addition to this information, which must certainly be present, you are free to add whatever you think belongs in a CV.

2.3 Technical Requirements

- HTML to build the structure of your page.
- CSS to define your page style.
- You split up your HTML and CSS**in separate files**!
- You may**NO**use CSS libraries, so you write all the CSS yourself!
- Inline CSS is labeled as incorrect.

NOTE: If you only add the technical requirements below and your website looks very basic or ugly, you will probably not succeed on the web project. The technical requirements below are mandatory to add but do not guarantee 10/20 on their own. Search the internet for beautiful designs and possibly animations to make your website beautiful. Proof in this project that you can independently look up and implement new learning material. After all, this is also what you will have to be able to do next semester in WPL 2 and next year during your internship. If you find something online that you would like to implement but you do not understand how it works, you can always ask specific questions. Always make it clear what you have already tried to understand by saying, for example, “I understand this piece of code, it works this way and that, but I don't understand this part. Can you explain to me what's happening here?”

Functional analysis

ID	Subject	Description
WEB-01	Page content	All information described under "minimum requirements" is available on the site.
WEB-02	Photo*hover*effect	If the user has the photo "*hovers*" a shadow appears behind the photo. This gives the impression as if the photo is floating.
WEB-03	Photo click	If the user clicks on the photo and holds down the mouse, the shadow created by the "*hover*" to appear. Instead, a border will appear around the photo.
WEB-04	Table alternating formatting	The rows in your table that contain data (not titles,*headers*or *footers*) are alternately given different formats. Even rows always look the same and odd rows have a second, different layout.
WEB-05	Work experiences list	Create an ordered or unordered (your choice) list where you list all your work experiences. Make sure you take care of this formatting nicely. Each item in the list must contain at least the following information: • Your role (function) • The company/organization • Period (from when to when you did the work) • Description of the work done
WEB-06	*Footer*	Create one at the bottom of the page*footer*that takes up the entire page width. Within this*footer*your name and class are on the left, the text "Assignment curriculum vitae" in the middle, and a "back to top" button on the right
WEB-07	Back upstairs-knob	If you click the "back to top" button in the*footer*is clicked, the browser scrolls all the way back to the top of your web page.
WEB-08	Layout, interactive elements, look & feel	The website should feel like a website to the user, NOT like a simple word document. So make sure your website feels interactive (e.g*hover*effects,*animations*, etc.). Use CSS to make your resume more attractive.
WEB-09	Language	The CV must be drawn up in Dutch or English.

2.4 Approach and tips

- Do the necessary research to use examples of multiple CVs as inspiration for the layout and style as well as for the components and structure. Great design examples:awwwards &dribble

- First make a simple sketch of the structure your CV will look like.
- You get points for having all the general information *look and feel* of your page/site and on implementing all tasks in the functional analysis. You are free to determine the layout of your site yourself, try to ensure that everything fits together nicely as much as possible.

- Prove yourself! After all, you have CSS at your disposal, so let your creativity run wild. For a website that looks like an ordinary white A4 sheet with some black letters, you will receive very few points. This does not mean that you should not use black and white anywhere, just make sure that you use colors and special formats and pseudoclasses in other places to make your page more interactive than a Word document.
- During the WPL 1 contact moments, feel free to ask how well you are doing in terms of look & feel if you have any doubts about whether your website is beautiful enough to succeed.

2.5 Submit assignment

This part of the project does NOT need to be submitted separately.

3 Part W.2 – Responsive web design

3.1 Objective

In addition to part W.1, you are expected to submit your CV*fully responsive*will make. This means that the design of the CV looks nice on any screen size and all texts are always clearly readable.

3.2 Technical Requirements

- For functional requirements around responsive design, only CSS or SCSS and media queries may be used. If you write SCSS, you must compile the file into CSS yourself and link it to your site.
- You split your HTML and CSS into separate files!
- You are NOT allowed to use CSS libraries, so you write all the CSS yourself!
- Inline CSS is labeled as incorrect.

3.3 Functional analysis

- Design - User interface
  - Ensure a professional website look and feel. To be
  - determined by the developer.

3.4 Functional Requirements

ID	Subject	Description
WEB-10	Uniformity	The choices of font style, font size, and color are applied uniformly throughout the document.
WEB-11	Mobile design	All information remains clearly legible. Also on mobile devices (screens smaller than 480px). Elements that do not fit next to each other are placed one below the other.

WEB-12	Mobile design	Hyperlinks and buttons on a mobile version remain easily accessible to click on (e.g. make them wider).
WEB-13	Mobile design	Elements that do not fit next to each other are placed one below the other.
WEB-14	Responsiveness	The CV retains a clear structure, even if the browser window is enlarged or reduced. Elements scale with the size of the screen (where logical) and do not overlap as the screen changes size.
WEB-15	External links	When the user clicks on the profile photo, he/she will be redirected to your LinkedIn page. If you do not have a LinkedIn profile, you may also refer to another social media platform. For example, Twitter account, YouTube channel, etc.
WEB-16	Table in mobile view	In the mobile design the table (desktop design) is no longer visible. The information (from the table) is displayed in a way that looks nice on a mobile device. The display method is freely selectable (but no table).
WEB-17	Table on larger screens	As soon as the screen is wide enough and the tablet or desktop design is shown, the table becomes visible again and the mobile variant disappears.

3.5 Approach and tips

- Media queries should be used in the CSS.
- Make sure that the correct design is shown depending on the width of the screen.

O	<= 480px	Mobile devices.
O	> 480px	Desktop and large screens.

3.6 Submit assignment

This part of the project does NOT need to be submitted separately.

4 Part C.1 – Basic version Cookie Clicker

4.1 Concept(s)

The project is based on the game Cookie Clicker. You can find an online version of this gamehere (https://cookieclicker.ee).

The aim of the game is to bake as many cookies as possible as quickly as possible. In the beginning you can only do this by literally clicking on the cookie. You generate one cookie per click.

The concept of the game is deceptively simple. As a programmer you are challenged to make clicking on a button as fun as possible. This relies on an important UX principle called gamification, which is used to get users and players more invested in an application. If you apply additional gamification techniques that fall outside the requirements of the assignment, you can do this**extra points**to score.

If clicking the button in your application is not fun, then your gamification is unsuccessful.

Gameplay:
The game starts by showing an image of a cookie. Initially your score is of course zero, but this can change by clicking on the image. Per (cookie) click, one cookie is generated and your score is increased by 1. While playing the game you will collect enough cookies to make investments. These investments are actually the essence of the cookie clicker: "What is the best investment for your cookies?"

End:
This game actually never ends. Every time you close the application, everything is reset and you start all over again.

4.2 Technical analysis

4.2.1 Technical requirements

- C# to create the logic of the application.
- XAML to build the UI.
- Visual Studio as a programming environment.

4.3 Functional analysis

4.3.1 Design – User interface

The application home screen may look like the following example*(see figure 1)*. However, you are encouraged to build your own design for your cookie clicker.*(Do not make a copy of the example.)*You can be inspired by the web game, but this is not mandatory.

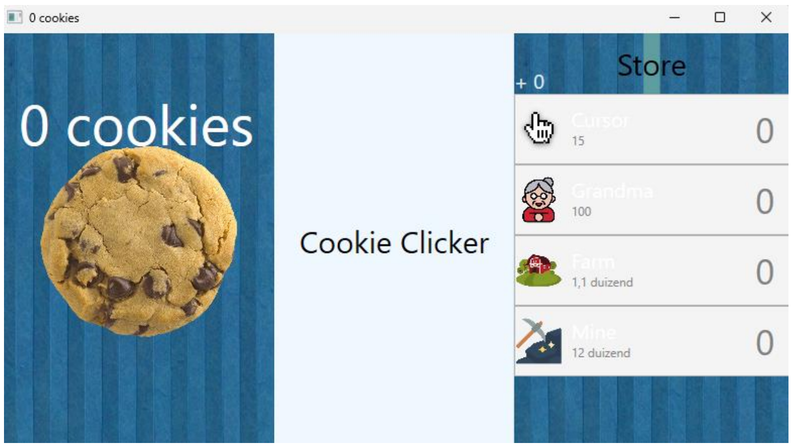


Figure 1: Home screen

As a player, I can click on the cookie image to collect cookies. The sum of collected cookies appears in the window title and in a text field above the image*(see figure 2)*. Once the player has enough cookie budget to make an investment, the buttons for the affordable investments are enabled*(see figure 2)*.

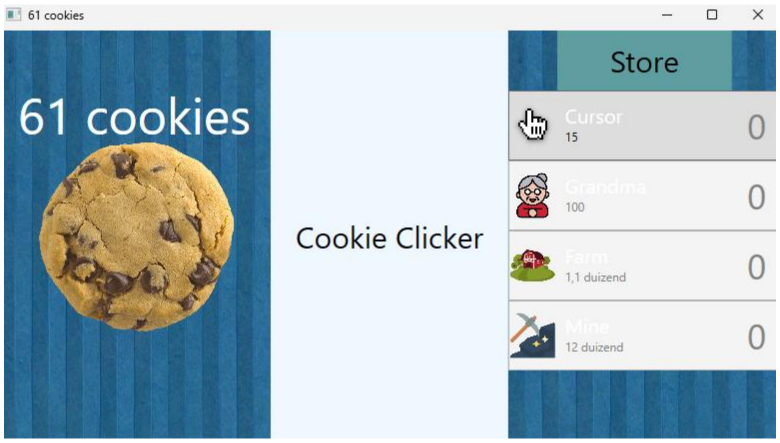


Figure 2: 61 cookies

The player can see the total amount of cookies he or she passively earns per second *(see figure 3)*.

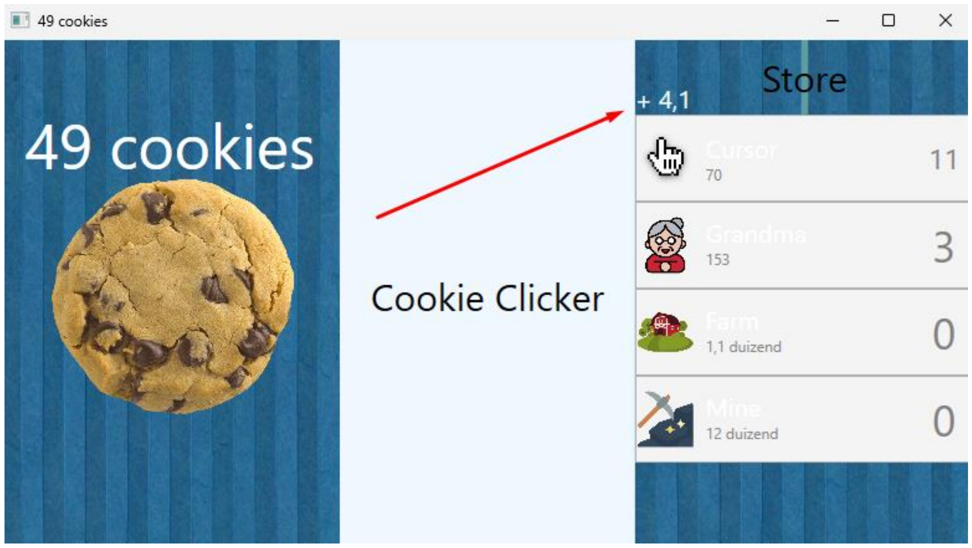


Figure 3: Passive income

The player can discover how much each investment yields per second in a tooltip that appears when hovering the cursor over an investment button*(see figure 4)*.

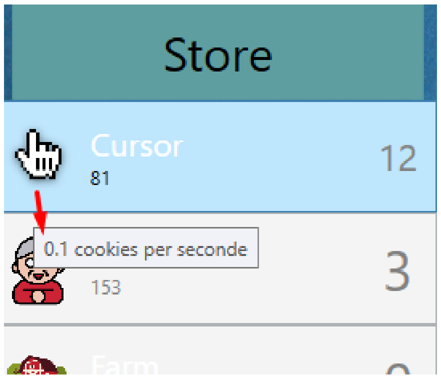


Figure 4: Tooltip with return on investment button

4.3.2 Functional Requirements

ID	Subject	Description
COOKIE CLICKER-01	Boot up	The application starts in an initial state where the score is shown at zero and an image of a cookie is visible. This image has a transparent background (for example, use .png).
COOKIE CLICKER-02	Background	Make sure the background of your application is filled with an image.*(Find out how to achieve this with an ImageBrush in XAML)*
COOKIE CLICKER-03	Click Cookie	As a player I can click on the cookie image to create a cookie. However, your cookie image is not a button and will not have a Click event handler.*(Tip: solve this with MouseDown and MouseUp events)*
COOKIE CLICKER-04	Click effect	As a player I can perceive a click effect by clicking the button. When the cursor is pressed on the cookie image, the image is reduced in size. When the cursor is released on the image, the image becomes larger again.*(Tip: You can use the properties ActualWidth and Width to easily achieve this effect.)*
COOKIE CLICKER-05	Click effect bugs	The click effect takes into account scenarios where the mouse button does not rise above the cookie image but outside it. Conversely, the scenario where the mouse button is pressed outside the image but released within the image is taken into account.
COOKIE CLICKER-06	Click score	As a player, I can click on the image to increase the cookie score by one.
COOKIE CLICKER-07	Cookie score	As a player, I can always see the total amount of cookies collected. Make sure the label floats above the clickable cookie. Each time a cookie is earned, this number must be updated immediately. Round the number of cookies down when you show it to the user, and of course store accurate decimal numbers afterwards.

COOKIE CLICKER-08	Cookie score title	As a player I can also find the cookie score from COOKIE CLICKER-07 in the title of the window.*(Tip: use methods for code that you have to write out two or more times)*
COOKIE CLICKER-09	Investments - buttons	As a player I see buttons in the game window with the different investments. Make sure that you can find at least the following investments on your screen: cursor, grandma, farm and mine.
COOKIE CLICKER-10	Investments – buttons disabled	Only the buttons for which the player has sufficient budget are enabled. As soon as a player has enough cookies for an investment, they are immediately enabled.
COOKIE CLICKER-11	Investments – buttons layout	As a player I can find information about the various investments in the layout of the button itself. Each button contains the following elements: • A transparent image representing the purchase. • The name of the investment. • The purchase price of the investment. • A tooltip indicating how many cookies per second this purchase yields. • How many times the player has already made the purchase.
COOKIE CLICKER-12	Investments – issue cookies	As a player, I can spend my collected cookies by clicking on a button of a specific investment to purchase that investment, such as: cursors, grandmas, farms, mines, factories and banks.
COOKIE CLICKER-13	Investments - price	The player can make investments that provide passive income from cookies. These investments become more and more expensive the more times the specific investment has already been purchased. • Use the following formula to calculate how much each purchase costs at any point in time: = ∗ 1.15 _ _ *h* • See**table 1**for an overview of all basic prices. • Round the price of each investment up, so that you can display nice integers on the screen. • *Tip: Use the Math class for this.*

COOKIE CLICKER-14	Investments – yield	Each investment has a passive income that is processed in the game every 10 milliseconds. Use a timer that starts when you start the game to update the total amount of cookies collected every 10 milliseconds. See**table 2**for an overview of all returns per investment.
COOKIE CLICKER-15	Investments – yield Tick	As a player I see each Tick updating the values in the window. Make sure that at least the following controls in the window are updated: • The title of the window. • The number of cookies in the text field hovering above the cookie image. • The price of every possible investment. • The quantity of each investment purchased. • The amount of cookies per second that the player earns.
COOKIE CLICKER-16	Cookies per seconds	Provide a creative way to indicate to the player how many cookies per second he or she earns. Keep gamification in mind: big numbers, colors, movement and sound.

4.3.3 Tables

Table 1: Overview of basic prices

Investment	Basic price
Cursor	15
Grandma	100
Farm	1100
Mine	12000
Factory	130000

Table 2: Overview of returns per investment

Investment	Yield per second	Yield per 10 milliseconds
Cursor	0.1	0.001
Grandma	1	0.01
Farm	8	0.08
Mine	47	0.47
Factory	260	2.60

4.4 Approach and tips

4.4.1 Step-by-step plan

- Thoroughly analyze what needs to be programmed and how you are going to do it.
- Try to figure out what the most difficult part of the exercise is (e.g. how do you create an image click event?) and investigate how to solve it.
- Program the game in Visual Studio.
- Test the game to see if it works in all conditions.
- Make sure your code is nicely structured (e.g. use meaningful names for variables).

4.4.2 Programming Tips

- Create a method that increases your score.
- Use private variables in the application to save your score.
- Use constants to store fixed values   about the investments.

4.5 Submitting an assignment

This part of the project does NOT need to be submitted separately. You will submit the final iteration on blackboard.

5 Part C.2 – First Cookie Clicker extension

5.1 Concept(s)

The first version of the game is ready. In this assignment we will focus on so-called **change requests**. We are going to expand the game, but also adapt it. The goal is to maintain the requirements from the previous iteration while implementing the new requirements. In iteration two we will visualize the passive income on the screen so that the player can admire his collection of purchases.

This iteration we are making the following adjustments to the concept of the game:

- The game will visualize the investments. For example, the player should see an image appear in the window for each investment purchase.
    - Use a StackPanel with a specific background per investment. Use a scroll
    - element to make all investments visible.
- Add the Factory, Bank and Temple to your game. You will find the corresponding tables in this chapter.
- Provide a name for the cookie bakery that the player can customize.
- Make sure that all numbers appear legibly and written in a combination of words and numbers on the screen.
- Provide a scalable user interface.

You will add WPF controls modularly from your code based on the code snippets you have received. You will use this, among other things, for the visualization of investments and for the upgrade store.

5.2 Technical analysis

5.2.1 Technical requirements

- C# to create the logic of the application.
- XAML to build the UI.
- Visual Studio as a programming environment.
- The new functionalities and change requests must be kept up to date with version management (via BitBucket, for example)! Create a branch for each iteration and use the issue keys in your commits.

5.3 Functional analysis

5.3.1 Design – User interface

We give you complete freedom to determine the layout of your extensions yourself. Keep the gamification concept in mind!

5.3.2 Functional Requirements

The additional requirements and/or change requests for the game are as follows:

ID	Subject	Description
COOKIE CLICKER-17	Big numbers	As a player I can perceive large numbers in a readable way. Numbers that one**price**or the**total amount of cookies**show and be greater than or equal to one million are written in a combination of numbers and words and rounded to three decimal places*(e.g.: 1,025,643 becomes 1.026 million)*. Make sure your application can use the following terms for numbers: o Million o Billion o Trillion o Billiards o Trillion
COOKIE CLICKER-18	Hundred thousand-numbers	As a player I can also perceive smaller numbers in a readable way. Make sure that numbers under one million but greater than a thousand have a space between hundreds and thousands*(e.g.: 15241 becomes 15 241)*
COOKIE CLICKER-19	New ones investments	Elaborate COOKIE CLICKER-12 further. As a player, I can spend my collected cookies by clicking on a button of a specific investment to purchase that investment, such as: cursors, grandmas, farms, mines, factories, banks and temples. See table 3 and 4.
COOKIE CLICKER-20	Bakery name	As a player, I can read the name of my bakery above my total amount of cookies in a**Label**. The name of my cookie bakery is “PXL-Bakery” by default.

COOKIE CLICKER-21	Bakery name modify	As a player, I can change the name of my bakery by clicking on it. Use a dialog box to confirm the change. Make sure that the user enters a valid name: it cannot be empty or consist of white space.
COOKIE CLICKER-22	Investments categories	As a player, I can see a category field in the window for each type of investment. These fields, if visible, will always appear on the screen in the following order: cursors > grandmas > farms > mines > factories > banks > temples. Make sure each category field has a background image large enough to represent all purchases.*(Tip: you can use the TileMode of an ImageBrush.)*
COOKIE CLICKER-23	Investments visualization	As a player, for each purchase I can see an image that represents my purchase. In other words, if I, as a player, purchased twenty grandmas, then twenty images of grandmas were added to the screen. Each purchase appears in its category. These category fields only become visible once at least one investment of that category has been purchased.*(Tip: see Code Snippets 3.3.4)*
COOKIE CLICKER-24	Scroll bars	As a player, I can use scroll bars to see investment buttons and images of investments that would fall out of the window. These scroll bars are only visible if the content does not fit on the window.*(Tip: you can use ScrollViewer for this)*
COOKIE CLICKER-25	Scalable UI	As a player, I can resize the window and still get a usable interface. Make sure that the text always remains readable in your window and that controls never overlap.*(Tip: it is best to use layouts to structure your window. Then you can make your controls larger by encapsulating it with a Viewbox.)*
COOKIE CLICKER-26	Investments visibility	As a player, I can only see the investment buttons of a category whose base price costs the same or less than the total amount of cookies I have ever collected.*(For example: After I have clicked on the cookie 15 times, only then does the cursor investment appear. After I have a total of one hundred cookies ever*

		collected (even if part of it has already been spent on investments), the grandma investment will appear.)

5.3.3 Tables

Table 3: Overview of basic prices

Investment	Basic price
Cursor	15
Grandma	100
Farm	1,100
Mine	12,000
Factory	130,000
Bank	1,400,000
Temple	20,000,000

Table 4: Overview of returns per investment

Investment	Yield per second	Yield per 10 milliseconds
Cursor	0.1	0.001
Grandma	1	0.01
Farm	8	0.08
Mine	47	0.47
Factory	260	2.60
Bank	1,400	14
Temple	7,800	78

5.3.4 Code Snippets

You can add controls in your WPF application via XAML and via C#. You could technically write your entire layout in C#. However, this is not very readable, easy and does not use the designer. You use C# to create controls when you want to dynamically adjust your layout. This allows you, for example, to generate additional buttons, text fields, layouts or images. Here are some examples of XAML code translated to C#.

XAML:

```
<TextBlock Name="TxtExample" Text="Example"></TextBlock>
```

C#:

```
TextBlock TxtExample = new TextBlock();

TxtExample.Text = "Example";
```

XAML:

```
<StackPanel Name="StckExample" Orientation="Vertical" Background="Black"
Grid.Column="1" Height="50" VerticalAlignment="Top"></StackPanel>
```

C#:

```
StackPanel StckExample = new StackPanel();

StckExample.Orientation = Orientation.Vertical;

StckExample.Background = Brushes.Black;

Grid.SetColumn(StckExample, 1);

StckExample.Height = 50;

StckExample.VerticalAlignment = VerticalAlignment.Top;
```

XAML:

```
<Button Name="BtnExample" Margin="10" FontSize="10" Click="BtnExample_Click">

Example </Button>
```

C#:

```
Button BtnExample = new Button();

BtnExample.Margin = new Thickness(10);

BtnExample.FontSize = 10;

BtnExample.Content = "Example";

BtnExample.Click += BtnExample_Click;
```

After you create a control in C#, you will still need to add it to a layout. If you don't do this, your control will exist but not be visible in the window. Below are two examples where the Button and TextBlock are added to a WrapPanel created in the XAML code.

You use this here**Children**property of a layout (StackPanel, WrapPanel, Grid, ...). You can add and remove controls using the**Add()** and**Remove()**method that you can call on the Children property.

XAML:

```
<WrapPanel Name="WrpExample" Margin="10"></WrapPanel>
```

C#:

```
WrpExample.Children.Add(BtnExample);

WrpExample.Children.Add(TxtExample);
```

5.4 Approach and tips

5.4.1 Step-by-step plan

- Thoroughly analyze what needs to be programmed and how you are going to do it.
- Try to figure out what the most difficult part of the exercise is (e.g. how do you create an image click event?) and investigate how to solve it.
- Program the game in Visual Studio.
- Test the game to see if it works in all conditions.
- Make sure your code is nicely structured (e.g. use meaningful names for variables).

5.4.2 Programming tips

- Use the Children property of a layout to dynamically add elements to your layout.

- Use data structures to store your variables, so that you bring more structure to your application. Make sure that you can easily add new investments in the future.

- Use constants to store fixed values   about the investments.

5.5 Submitting an assignment

This part of the project does NOT need to be submitted separately. You will submit the final iteration on blackboard.

6 Part C.3 – Second Cookie Clicker extension

6.1 Concept(s)

Fine-tuning is planned in this iteration. Make the following adjustments to the game concept:

- Provide a bonus store. Depending on the investment, this will double the amount you earn on cookies. For example, 100% extra income per bonus for a specific investment.

- Animate a cookie falling from the cursor or from the top of the window to the bottom for each cookie that is generated. Limits this to a maximum of 50 cookies on the screen at a time.
- Golden cookie: create an event that makes a golden cookie appear on the screen at a random location every minute. If clicked, the player will receive 15 minutes worth of cookies (depending on his/her cookie production).
- Provide a quest: this shows an achievement on the screen that indicates the progress of the game.

Quest	Notification
reach 20 cookies per second	Your shop is starting to take off and you are attracting people from all over the village to your bakery
buy 10 grandmas	In the evening you organize bingo games to attract grandmothers from all over the village to your bakery. The free worker is a successful business model.
. . .	. . .

Come up with a total of 20 assignments/quests that the player can complete. Have the notification appear in a message box. The player must be able to find the completed quests and the associated notifications in the window. Provide a layout that can clearly visualize your notification history.

6.2 Technical analysis

6.2.1 Technical Requirements

- C# to create the logic of the application.
- XAML to build the UI.
- Visual Studio as a programming environment.
- The new functionalities and change requests must be kept up to date with version management (via BitBucket, for example)! Create a branch for each iteration and use the issue keys in your commits.

6.3 Functional analysis

6.3.1 Design – User interface

We give you complete freedom to determine the layout of your extensions yourself. Keep the gamification concept in mind!

6.3.2 Functional Requirements

The additional requirements and/or change requests for the game are as follows:

ID	Subject	Description
COOKIE CLICKER-27	Bonus store - knob	This bonus will always double the income from an investment. For example: when you buy a bonus for the grandmas, you double the income of all grandmas. Provide a bonus button on the screen that appears as soon as the respective investment button is visible. The button shows the cost of the bonus and the current multiplication (x1, x2, x4, x8, etc.). Make sure your button fits into your current layout. Furthermore, the button is only enabled if the player can pay the price of the bonus (see COOKIE CLICKER-28).
COOKIE CLICKER-28	Bonus store - Prices	To calculate the correct price of the bonus, we are inspired by the web version (see table 5). For the cost price you use the following logic: - the first doubling is base price * 100 - the following is base price *500 - then basic price * 5,000 - then base price * 50,000 - then base price * 500,000 - etc. The base price here is the basic price of each respective investment.
COOKIE CLICKER-29	Golden cookie	Create an event that has a 30% chance every minute to make a golden cookie appear on the screen at random

		place. If clicked, the player will receive 15 minutes worth of cookies (depending on his/her cookie production).
COOKIE CLICKER-30	Quest	As a player, I can read a notification in a message box every time I complete a quest. Provide at least 20 hidden assignments for the player. When the player has completed one of your assignments, a text related to your assignment will appear in a message box. See table 6 for an example of quests.
COOKIE CLICKER-31	quest - history	As a player I can read old notifications of the hidden quests in a history. Provide a place in your layout where the player can scroll through a list of completed quests. The history contains the quest and notification in the order in which they were achieved.

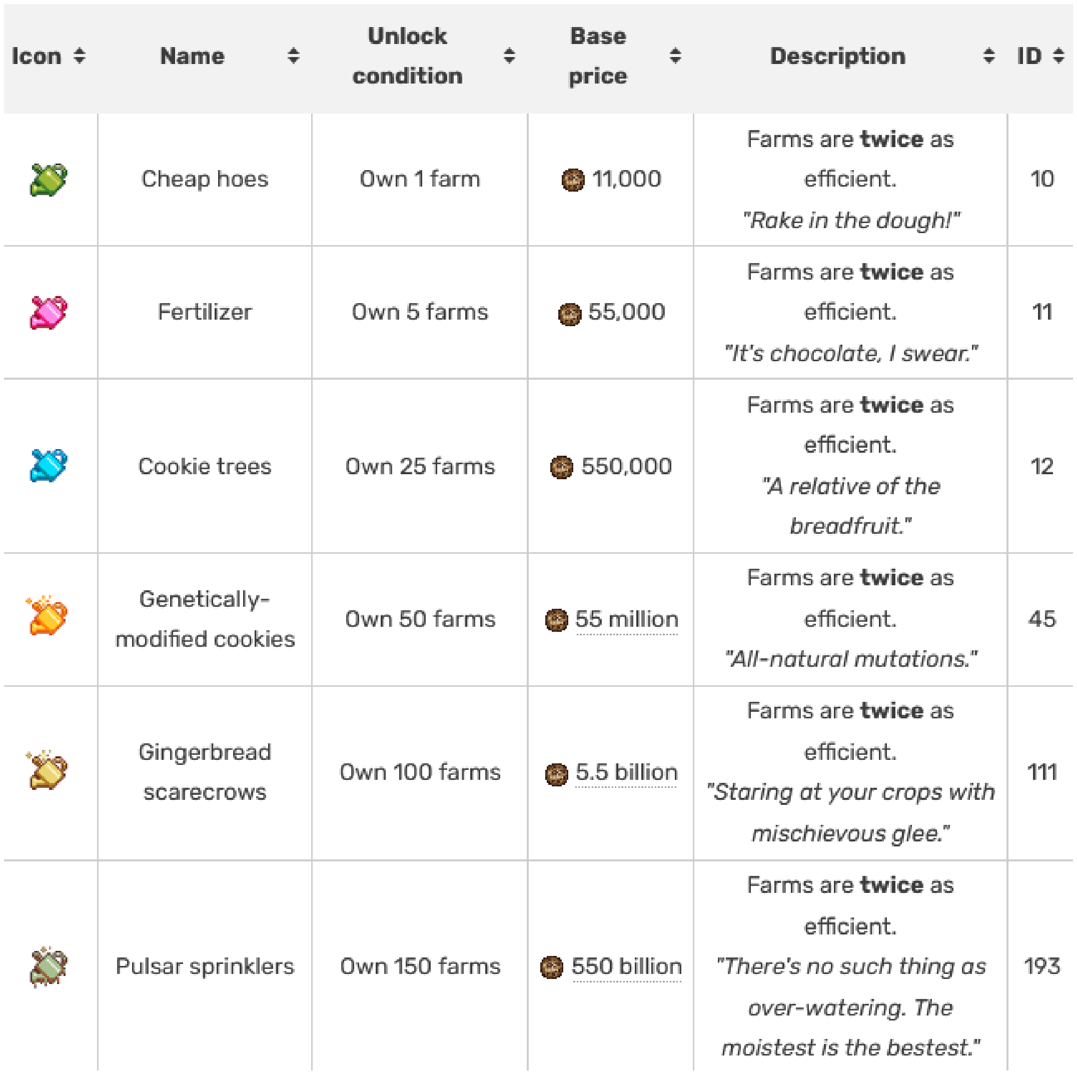
6.3.3 Bonus Requirement

The requirement below is not mandatory to implement. However, if you implement them successfully, you can earn a bonus point.

COOKIE CLICKER-32	Animation	Create an animation to drop a cookie from the cursor or from the top of the window for each cookie that is generated. Limits this to a maximum of 50 cookies on the screen at a time.

6.3.4 Tables

Table 5: Example bonus store (Use the pattern from COOKIE CLICKER-28 to calculate the prices of the different investments. You should not take into account an “Unlock condition”)



Icon	Name	Unlock condition	Base price	Description	ID
	Cheap hoes	Own 1 farm	11,000	Farms are **twice** as efficient. *"Rake in the dough!"*	10
	Fertilizer	Own 5 farms	55,000	Farms are **twice** as efficient. *"It's chocolate, I swear."*	11
	Cookie trees	Own 25 farms	550,000	Farms are **twice** as efficient. *"A relative of the breadfruit."*	12
	Genetically-modified cookies	Own 50 farms	55 million	Farms are **twice** as efficient. *"All-natural mutations."*	45
	Gingerbread scarecrows	Own 100 farms	5.5 billion	Farms are **twice** as efficient. *"Staring at your crops with mischievous glee."*	111
	Pulsar sprinklers	Own 150 farms	550 billion	Farms are **twice** as efficient. *"There's no such thing as over-watering. The moistest is the bestest."*	193

Table 6: Example quests and notification

Quest	Notification
Reach 20 cookies per second	Your shop is starting to take off and you are attracting people from all over the village to your bakery.
Buy 10 grandmas	In the evening you organize bingo games to attract grandmothers from all over the village to your bakery. The free worker is a successful business model.
Reach 100 cookies per second	Your cookie fields are infamous. You appear every day in the newspaper about your famous cookie plants that you grow in your fields.
Buy 10 factories	The flesh is weak, You throw all the workers at the door. As of today, your factories are complete automated.

6.4 Submitting an assignment

This part of the project does NOT need to be submitted separately. You will submit the final iteration on blackboard.

7 Submit assignment

7.1 Web application

For the web application you need to create a folder containing the following files:

- All HTML and CSS files (part 1 & 2)

Place this folder in a zip file with the name**lastname_firstname_web.zip.**

You ensure that the references (“hyperlinks”) to your files work. The folder is opened by the lecturer as a WebStorm project, and all the “hyperlinks” should work immediately.

7.2 C# command

Create a zip file with the name**lastname_firstname_csharp.zip**containing the following elements:

- Visual Studio Solution with ALL code from parts 1, 2 and 3.
- You can remove the “bin” and “obj” from your project.
- Add one**link**to you**git repo**in the description of your entry and send an invitation tosander.depuydt@pxl.be andkoen.bloemen@pxl.be

When unpacking and opening the Visual Studio solution (by the lecturer), it must compile, start and work error-free.

7.3 Important tip

Before uploading your assignment to BB, distribute your solution (C# and Web) to one of your fellow students. And check on his/her laptop whether:

- The Visual Studio solution for CookieClicker can be opened.
- The Visual Studio solution for CookieClicker compiles without error.
- The Visual Studio solution for CookieClicker works as it should.
- The web page for the CV can be opened from WebStorm.
- All elements in the CV are displayed correctly.
- Images are displayed correctly.

Please note: you do not copy a code from a colleague!

7.4 Location and deadline

Both zip files must be uploaded to the designated location on BlackBoard. The coaches will provide the necessary instructions for this.

C# project deadline: 07/01/2024 11:59 PM